THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 62, No. 1, January 1983
Printed in U.S.A.

The 3B20D Processor & DMERT Operating System:

3B20D Craft Interface

By M. E. BARTON and D. A. SCHMITT

(Manuscript received March 10, 1982)

The 3B20D craft interface package includes hardware, firmware, and software that enables telephone company craftspeople to obtain the status of and exert control over the system. Because this package consists of one or more standard keyboard-display terminals for human-machine interactions, it is flexible and can be adapted to a broad variety of applications. Furthermore, the use of standard terminals and data link protocols allows for inexpensive remote access with capabilities similar to local access capabilities. Finally, the use of video displays has made it possible to provide easy-to-use menus that guide the craftspeople through some of the complex control operations. This article describes the 3B20D craft interface capabilities and the internal architecture of the package.

I. INTRODUCTION

The "craft interface" is that part of the 3B20D Processor that enables people to obtain status information and exert control over the system. To those not involved in telephony, the word "craft" may seem odd. It has traditionally been used to refer to the people who work in and around telephone switching offices performing various maintenance functions on the equipment. In this article, the term is used somewhat liberally to mean any person who interacts with the 3B20D to perform administrative and maintenance functions.

The 3B20D's craft interface is a marked departure from previous systems developed at Bell Laboratories because it relies almost exclusively on video displays and keyboard controls instead of the key-lamp panels and teletypewriters usually found in the Master Control Center (MCC) of electronic switching systems. Status information is presented visually as graphical displays and text messages on various terminals and printers. There is also a capability to provide audible status by connecting the 3B20D to an audible alarm circuit. System control is exerted primarily via a keyboard attached to the video display terminal, although the 3B20D also includes a separate power control panel for each major hardware unit.

Another important enhancement lies in the ability to access and control all aspects of the system from remote locations such as Switching Control Centers (SCCs). In the past, remote access was obtained by "piggy-backing" data links onto the typewriter terminals in the telephone office and by connecting a telemetry unit to the key-lamp control panel. The 3B20D has introduced a more "intelligent" data link using the CCITT X.25 communication protocol. This link can carry considerably more information and is less vulnerable to noise and other data communication failures. Furthermore, the use of the international standard message protocol (X.25) will standardize remote access to the 3B20D via packet switching networks.

This article first provides an overview of the 3B20D craft interface, primarily concentrating on how the system appears to the craftspeople. Then the internal architecture is described and the various 3B20D applications usages of the general facilities provided in the common system are explained.

II. OVERVIEW

This section describes the 3B20D craft interface as it appears to the people who use it to administer and maintain the system.

2.1 Hardware

The most frequently used parts of the craft interface are shown mounted in two equipment frames in Fig. 1. The left frame contains a "read-only printer" or ROP* on which all important status messages are logged. The right frame contains a keyboard-display terminal that is commonly referred to as the "maintenance CRT," or MCRT. Telephone switching applications of tbe 3B20D can choose either a frame-mounted or desk-mounted arrangement for the ROP and MCRT. A desk mounted version is shown in Fig. 1.

2.2 Text messages

One way in which the 3B20D communicates with the craftspeople is via text messages. For example, when the message

* From the viewpoint of a programmer, it is a "write-only printer," since the programmer can only send (i.e., write) messages to it. However, the craftsperson cannot type on this device, and so from that viewpoint it is a "read-only printer."

DGN:CU 0;UCL!

is typed, the central processing unit 0 (CU 0) is diagnosed. The UCL keyword indicates that the diagnosis is "unconditional," which means that all tests will be run even if some of the early tests fail. When the diagnosis is complete, the CU diagnostic prints a text message such as:

DGN CU 0 COMPLETED ATP

This means that the diagnosis has been completed and all tests passed (ATP). For initial 3B20D applications, the text messages conform to the Bell System craft interface syntax, commonly known as the Program Documentation Standard (PDS) Language. However, all new switching systems developments will be adopting a craft interface language sanctioned by the International Telegraph and Telephone Consultative Committee (CCITT) under the name MML. Since PDS and MML are similar, and since the 3B20D is expected to enjoy broad use in international applications, the operating system was designed so that each application can easily choose the appropriate syntax.

Text messages are typed on the MCRT keyboard, and the response messages are displayed on the MCRT video display and/or printed on the ROP. The basic repertoire of messages available with the 3B20D covers a broad range of maintenance and administration activities. Each application can easily add its own messages to this repertoire.

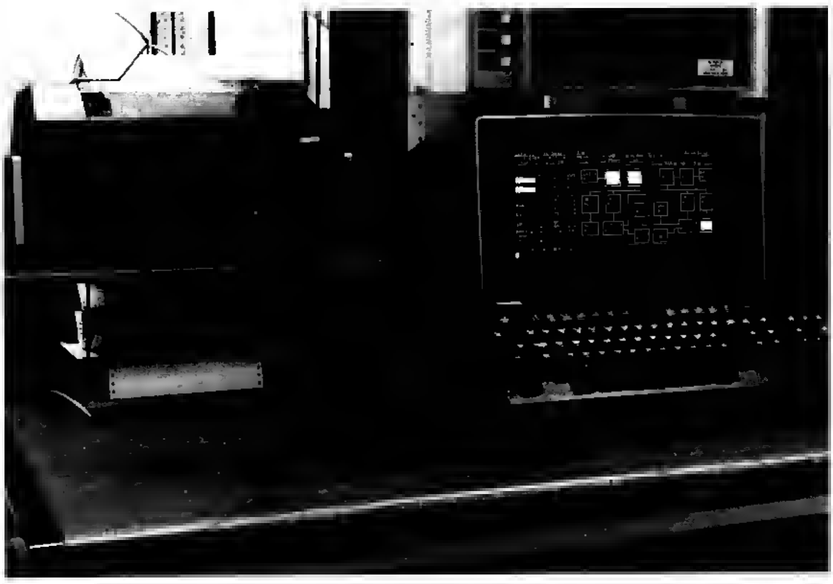

Fig. 1—Craft interface printer and terminal.

2.3 Control and display functions

As mentioned earlier, previous systems used key-lamp panels to display system status and to receive control signals from the craftspeople. The 3B20D uses the MCRT video screen and keyboard, as shown in Fig. 2, in place of such a panel. The upper part of the screen always contains a summary of important system indicators, including CRITICAL, MAJOR, and MINOR severity alarms and "type" alarms, such as CU and BLDG/PWR, which is the indicator for building power. The middle part can display a variety of "pages" that show system status in a graphical form. Finally, the lower part of the screen is used for text input and output.

The standard 3B20D software includes several display pages related to the common processor equipment, and each application can easily add its own pages. The "Common Processor Display Page" shown in Fig. 3 provides a diagram of the redundant components in the basic processor complex. At the left of the diagram is a "menu" listing the control operations that can be invoked when this page is displayed. To select a menu item, the craftsperson depresses the CMD/MSG key, which switches the craft interface from text message mode to command

Fig. 2—Craft interface video screen and keyboard.

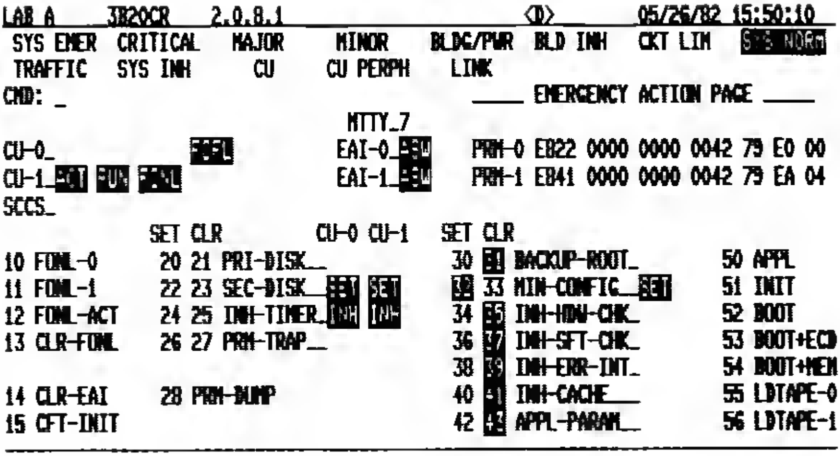

Fig. 3—Emergency action interface display page.

mode. Then the craftsperson types the menu item number, replacing "x" with a 0 or 1 where necessary. Finally, depression of the RETURN key (or the ! key) causes the command to be executed.

The craft interface stays in the command mode until the CMD/MSG key is depressed again. This key is one of four "special function keys." The ALM RLS key is used to retire audible and visual alarms. The EAI DISP key places the craft interface in the emergency action mode, which is described below. When in EAI mode, the NORM DISP key returns the screen to its previous display.

The Emergency Action Interface Page is different from other pages because it is directly controlled by a microprocessor in the MTTY controller (MTTYC) and, therefore, can be used even when the 3B20D software is not operating. As shown in Fig. 4, this page contains menu items that enable the craftsperson to re-initialize the system or to force the redundant units into a particular configuration. Typically, this page is used only when system sanity is suspect.

2.4 Remote access

All capabilities of the craft interface except the power control panels can be accessed from a remote maintenance center via a dedicated data link that is attached to the MTTYC. The standard arrangement includes a primary and a backup link, both of which use the CCITT X.25 communication protocol. The remote site is usually a Switching Control Center (SCC) that contains a collection of computers and terminals that interface with these X.25 links and provide the SCC craftspeople with sophisticated analysis and maintenance tools.

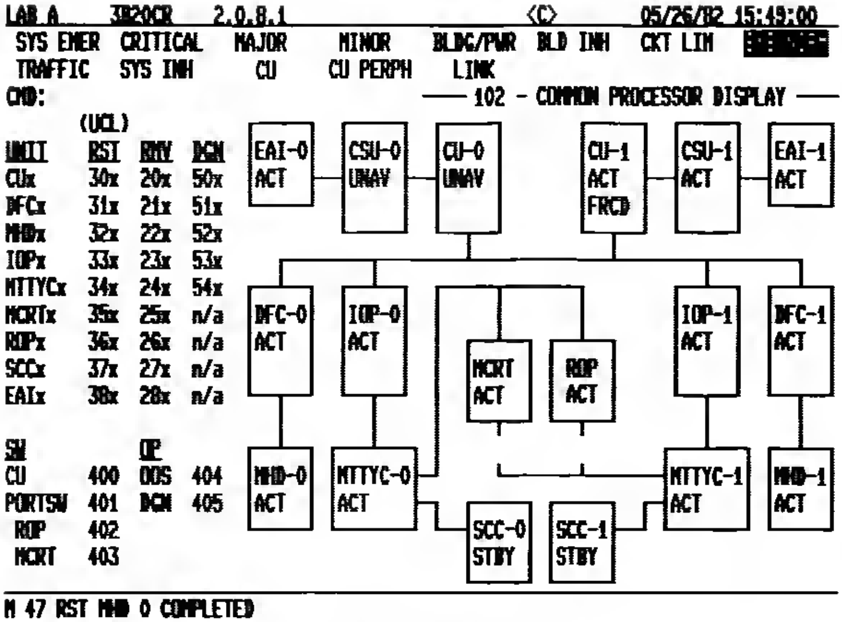

Fig. 4—Common processor display page.

III. CRAFT INTERFACE ARCHITECTURE

This section discusses the hardware and software architecture of the 3B20D craft interface. Figure 5 shows the arrangement of hardware units pertinent to the craft interface, while Fig. 6 shows the software modules. The discussion of the hardware architecture that follows will cover the I/O Processor (IOP) driver and MTTYC handler software, as they are the fundamental parts of DMERT required to access the hardware.

3.1 Hardware architecture

Referring to Fig. 5, one sees that each of the duplex processors is connected to both IOPs, and that each IOP supports up to sixteen peripheral controllers (PCs). Various PCs exist for terminals and printers, data links, tape units, etc. The IOP driver process, which is the part of DMERT responsible for communication with the IOPs, contains "handlers" that deal with the specialized functions of the PCs. The following handlers are pertinent to the craft interface: craft interface handler, X.25 handler, emergency action interface handler, general-purpose terminal handler, and alarm handler.

3.1.1 Craft interface handler

The MCRT, ROP, and X.25 links are attached to a PC known as the maintenance teletype controller, or MTTYC. The craft interface

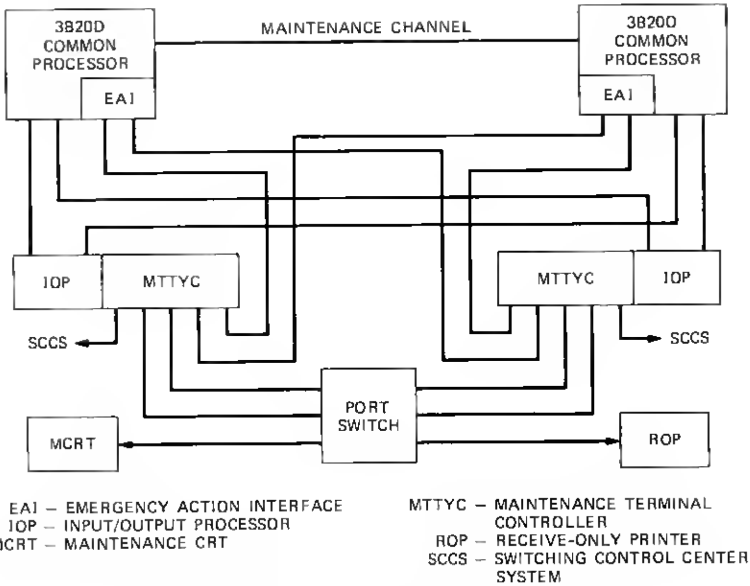

Fig. 5—Craft interface hardware overview.

handler controls the transfer of data to and from the peripheral devices associated with the MTTYC. The MCRT and ROP are administered directly by the MTTYC, while the X.25 links require the additional services of the X.25 handler described later. For each device or data link attached to the MTTYC, the handler supports all standard access operations of the *UNIX** operating system. In addition, this handler treats the single MCRT terminal as two "virtual terminals," with the upper part of the screen used for control/display functions and the lower part used for text messages as shown in Fig. 2. Each virtual terminal appears to the higher-level software as a separate device.

3.1.2 X.25 handler

The X.25 handler provides communication with a remote maintenance center via 1200 to 9600 bits per second synchronous data links using levels 2 and 3 of the CCITT X.25 protocol. Level 2, referred to as the link layer, provides link initialization, error control, and flow control on the physical data link and is implemented as firmware within the MTTYC. Level 3, referred to as the packet layer, multiplexes several independent data streams (logical channels) on the physical link and is implemented by the X.25 handler software. In effect, the X.25 handler makes the single MTTYC look like a multitude of independent communication channels.

* Trademark of Bell Laboratories.

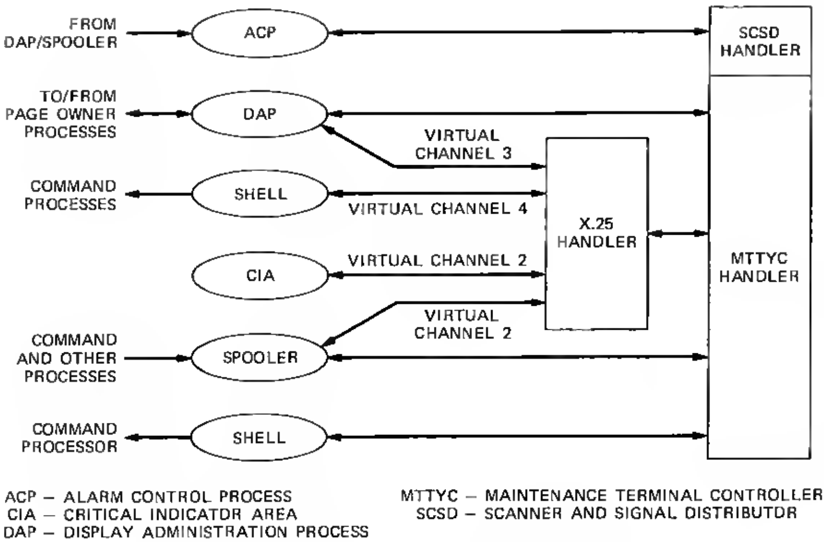

Fig. 6—Craft interface software architecture.

3.1.3 Emergency action interface handler

The Emergency Action Interface (EAI) is a processor circuit pack that provides basic status information and manual control even under extreme circumstances. That is, the EAI circuit gives craftspeople limited access to the 3B20D regardless of DMERT software sanity. This access is in the form of the EAI page display shown in Fig. 4, which is controlled totally by the firmware in the MTTYC. The MTTYC interacts with the EAI circuit via the connection shown in Fig. 5 to acquire the status information for display on the MCRT. Also, when the craftsperson selects a menu item from the EAI display, the MCRT delivers the corresponding commands to the EAI circuit.

The emergency action interface handler only comes into play when DMERT is operating sanely. It has two major functions. First, it periodically "punches in" with the EAI circuit to indicate that the software is operating correctly. If the EAI (and, subsequently, the MTTYC) fails to receive this periodic signal, it will automatically initiate a system recovery operation to restore software sanity. Second, the EAI handler receives some non-emergency commands from the MTTYC via the EAI, including non-emergency initialization and reconfiguration signals.

3.1.4 General-purpose terminal handler

The craft interface subsystem supports terminals other than the MCRT and ROP via the teletype controller, or TTYC. This controller offers all MCRT and ROP functions except access to the EAI circuit. Its handler is similar to the craft interface handler used with the MTTYC and, in particular, offers the dual virtual terminal mode of operation needed to intermix control/display functions and text functions on a single terminal. This capability is typically used for dial-up monitoring of a system from a Western Electric or Bell Laboratories product support center.

3.1.5 Alarm handler

The scanner and signal distributor (SCSD) peripheral controller provides sense and control points that are tied into the system power controls and alarms. The SCSD handler detects sense point state changes and sends commands to change the states of control points. Higher-level software uses these capabilities to detect situations such as power removal, fuse operation, or thermal warnings and to respond by activating audible alarms or power shutdown circuits. The application can also tie into the SCSD and configure the higher-level software to detect and react to such things as building intrusion alarms.

3.2 Software architecture

Figure 6 shows the modules that comprise the standard DMERT craft interface software subsystem. Already discussed were the device handlers that connect to the modules on the right of the figure. Each application usually adds its own modules that tie into the interfaces on the left. Also, many other parts of DMERT (e.g., the diagnostic subsystem) connect to the craft subsystem via these front-end interfaces. The modules in the middle of the figure are the "workhorses" of the craft subsystem and provide the internal interfaces used by DMERT programmers to interact with the craft personnel.

3.2.1 Text input processing (shell)

The Shell is the module that interfaces between the handlers and the processes that respond to text input commands. The term "shell" is borrowed from the *UNIX* operating system, and DMERT's craft shell operates in a manner similar to the other shell. That is, the DMERT shell reads an input line, parses it into a command verb and a list of "tokens," searches for the command process in the appropriate disk directory, creates the command process, and passes the list of tokens to it. The command process has access to a "shell library" that includes functions to do further parsing of the tokens.

The major difference between the DMERT shell and the other shell is that DMERT must parse commands that are typed in either the PDS or MML syntax, where as the *UNIX* operating system shell uses a more general syntax for a broader variety of applications.* Another difference is that the PDS and MML languages include the notion of a "locking acknowledgment." That is, an input command process is required to give a two-character response (e.g., OK if the command is successful, PF if a printout follows) within a few seconds after the person types the message, and no other command can be typed until the acknowledgment appears. In the *UNIX* operating system, message acknowledgments are not required and command type-ahead is allowed. Therefore, the DMERT sbell library includes functions that pass the acknowledgment back to the handler in order to unlock the terminal.

Referring again to Fig. 6, note that each text input channel has its own instance of the DMERT shell. This allows each channel to operate independently of the others, whicb means that several craftspeople can simultaneously interact witb the system.

3.2.2 Text output processing (spooler)

The DMERT output spooler accepts text strings from higher-level processes and directs them to the appropriate output devices. [The term "spooler" is a computer science anachronism that comes from the days when information waiting to be printed was temporarily stored on reels (spools) of magnetic tape.] One might ask why the higher-level process doesn't write directly to the device (via the device's handler, of course). There are two reasons to avoid direct writing.

First, the PDS and MML languages require that each message be enclosed in an "envelope" that clearly delineates the message. This envelope generally contains a time stamp, a message priority/alarm indicator, and an end-of-message delimiter. The time stamp can be as simple as the number of minutes past the current hour, or it can be the complete date and time. The priority/alarm indicator shows whether the message is a result of a manual or an automatic action and whether the action being reported requires immediate attention. Finally, the end-of-message delimiter provides for automatic logging and browsing of messages by a computer in the remote maintenance center. Centralizing the generation of the output message envelope in the spooler simplifies the work that higher-level processes must do to produce text output. Also, changes or additions to the envelope can be introduced easily by modifying only the spooler.

The second reason for using the spooler approach is that many messages must be sent to several places. For example, the usual mode

* For DMERT applications that require the more general *UNIX* shell on terminals other than the MCRT, it is possible to configure the system in such a way that the *UNIX* shell is automatically activated on some or all general-purpose terminals. In other words, both the DMERT and the *UNIX* operating system shells are compatible with the general-purpose terminal handler described earlier.

of operation when a remote maintenance center is attached via the X.25 links is to send every output message to both the MCRT, ROP, and the remote center. However, if the ROP runs out of paper or if there are no craft personnel near the 3B20D, messages can be routed only to the remote center. To handle these and the diverse message routing situations that can arise, the spooler maintains a "map" showing where messages are to be routed based on their priority/alarm indicator and on a message type code that is received from the process that generated the message. The map also can be configured to route some message types to disk files instead of or in addition to printing them. This feature is useful for keeping a log of messages that are sometimes needed for problem analysis but that would overload the ROP or X.25 links if sent routinely. Input commands are provided to print the contents of these logging files when needed.

3.2.3 Control/display processing

The Display Administration Process (DAP) administers the upper part of the MCRT (and, possibly, other video terminals) containing the displays that replace the traditional key/lamp panels for 3B20D applications. DAP's fundamental purpose is to display "pages" from its repertoire and to accept commands listed on "menus" associated with the pages. Figure 3 shows the Common Processor Display Page, which is one of the standard pages delivered with DMERT. Typically, the majority of display pages are defined by the specific application processes.

For each page, there is a Page Description File (PDF) containing a pseudo-program that describes how the page should be "painted" on the video screen and what menu selections are allowed. PDFs are constructed like programs and compiled by a page description file generator (PDFGEN) program.

3.2.3.1 Display functions. When DAP begins execution, no pages are active. Then, as the various parts of DMERT and the application are initialized, they send interprocess messages to DAP requesting that specific PDFs be loaded into main memory and activated as display pages. A maximum of 64 pages can be active at any one time. Other interprocess messages tell DAP which of the active pages to display on each video terminal.

Each page consists of up to 128 graphical constructs known as "indicators," a term reminiscent of the lighted indicators on a key/lamp panel. The process that initially informs DAP to activate a page becomes the owner of that page and it and other processes can subsequently inform DAP (via interprocess messages) to change the states of the indicators. For example, one popular type of indicator is the rectangle enclosing a few text items, such as the CU-0 box in Fig.

3. The CPDP page owner can send messages to DAP causing the phrase UNAV to change to OOS when control unit 0 (CU-0) changes from the unavailable state to the out-of-service state.

These state changes can be communicated in detail, for example, by sending a message to DAP specifying the characters OOS to replace UNAV. However, the usual method is to use state numbers instead of the actual characters. DAP has access to a table of 256 state entries specifying the standard text and video attributes associated with each of the 256 possible indicator states. The standard maintenance states, such as active, standby, and out-of-service, have predefined state numbers, and each application can define additional states for its own needs. The advantage of using state numbers is that the text and video attributes for each state can be centrally controlled. For example, one application could use the text ACT for the active state while another application used ACTIVE, and the only difference would be in the state table definition entries.

Video attributes were mentioned above in addition to the text that can be associated with an indicator. For the usual black-and-white terminals, DAP recognizes the "blink" attribute and the "reverse" attribute. The conventional use for the reverse attribute is to show that an indicator is, in some sense, active. In otber words, a reversed indicator is similar to a lit lamp on a key/lamp panel. The blink attribute is used to draw attention to a situation that requires immediate action, just like a flashing lamp. In Fig. 3, the SYS NORM indicator is reversed to show that the system is operating normally. If a major alarm occurs, the MAJOR indicator will blink until the ALM RLS key is pressed.

DAP also includes the capability to deal with color terminals, which have a much richer set of attributes. For example, the MAJOR indicator could be displayed as white characters against a red background, while the MINOR indicator would be white against yellow. It is possible to define indicator states in the most general way for color terminals and then have the Equipment Configuration Database contain MTTYC or TTYC options that "downgrade" the color states for black-and-white terminals. In the example mentioned above, both the red and yellow backgrounds would be mapped into the reverse attribute.

3.2.3.2 Control functions. In addition to displaying pages on the MCRT screen, DAP also can receive menu commands typed on the MCRT keyboard. These commands usually are referred to as "pokes" since they are similar to the action of poking a key on a key/lamp panel. As mentioned earlier, depression of the CMD/MSG key switches the terminal from the message mode to the command mode and vice-versa. When in the command mode, DAP displays a CMD: prompter towards the top of the screen, as shown in Fig. 3, and positions the cursor so that the typed characters appear in that line.

A command line consists of a number (usually 3 or 4 digits long) that optionally can be followed by some text characters. The craft interface handler, knowing that the terminal is in command mode, routes this input to DAP instead of to the SHELL. DAP examines the number to determine if it corresponds to a local or a global menu item. Local menu items are associated with the page(s) currently being displayed on the video screen. Global items are associated with any active page, even if the page is not currently displayed. In other words, a globally defined item will always be accepted and acted upon, even if its page is not being displayed.

If DAP successfully locates the menu item corresponding to the number that was typed, it usually sends an interprocess message to the owner of the page defining that menu item. This message contains the item number and the additional text characters typed, if any, as well as the originating terminal identification. The owner then takes whatever action is appropriate.

We used the word "usually" above because in some cases the response to a command is some simple action such as flipping to a new page on the display. In other cases, the PDF can specify a function to be executed by DAP upon receipt of the command, thereby bypassing the overhead of interprocess messages. One interesting aspect of this feature is the ability for DAP to translate a menu command into a text message to be passed to an instance of the SHELL, with the additional characters substituted in the message. This makes it possible for an application to design easy-to-use menus as an alternative to text message input, but to handle all terminal inputs internally as if they came through the SHELL.

A final note on commands has to do with locking acknowledgments. As with the SHELL, DAP requires a positive response to each command before another command can be accepted. For a command passed to a page owner via an interprocess message, the owner must send an acknowledgment message back to DAP within a certain time period or be abandoned. For commands handled via the function call approach, the function returns an acknowledgment code to DAP.

3.2.4 Forms processing

As described above, DAP is typically used for control/display functions related to maintenance activities such as configuration control. However, some applications require more general display functions than the indicator/menu approach appropriate for these maintenance scenarios. The craft subsystem provides facilities for entering textual information as part of displays.

A DAP page can be defined to contain input areas other than the standard command input line. When such a page is displayed, depression of the CMD/MSG key places the cursor at the first input area instead of at the command line. The craftsperson can enter text into this area and/or move the cursor, using the terminal cursor control keys, to the next input area on the page. When the RETURN key is hit, DAP passes the typed information to the page owner via an interprocess message.

3.2.5 Alarm processing

The Alarm Control Process (ACP) is the part of the craft subsystem responsible for sounding audible alarms and displaying a summary of current system status at the top of the video screen. ACP is created during DMERT initialization and notifies DAP to activate the page known as the System Summary Area (SSA). It also attaches itself to the SCSD handler to gain access to the signal distributor points used to sound audible alarms. The plant measurements data base is automatically updated for severity-type alarm counts.

As the spooler and DAP receive messages from the higher-level processes, they check for situations requiring audible alarms and/or changes in the system status summary. For the spooler, alarm information is contained in the message prefix received from the higher-level process. For DAP, this information is derived from the indicator state data. Both cases result in messages being sent to ACP, which then operates the signal distributor points via the SCSD handler and/or sends DAP messages to change the states of indicators on the SSA page. Application processes also send messages directly to ACP for alarms.

Another message received by ACP from DAP is a notification that the ALM RLS key has been depressed. This causes ACP to reset the signal distributor point controlling the audible alarms. This key depression is also reported from the MTTYPC directly to the SCSD audible alarm retire scan point.

The Critical Indicator Area (CIA) process is closely related to ACP and DAP. Its function is to extract from the SSA page sixteen "critical indicators" of system status and periodically send them to the remote maintenance center via the X.25 link for alarming and display. The remote center can use this periodic "heartbeat" from the CIA process as one test that the system is operating sanely.

3.2.6 Common processor display page

Thus far we have described the general hardware and software modules that comprise the 3B20D's craft interface subsystem. DMERT also includes several processes that are, in effect, users of the craft subsystem. One of these is the process that owns the Common Processor Display (CPD) page that we have frequently used for examples (see Fig. 3). This Real-Time Status (RTS) process is created as part of the DMERT initialization sequence and immediately sends messages to DAP to activate the CPD page. RTS also attaches itself to the Equipment Configuration Data Base (ECD), which it periodically examines to determine if any units shown on the CPD page have changed state. Spontaneous equipment configuration changes are reported to RTS through a library interface (CONFIG) from the various device handlers. In either case, appropriate messages are sent to DAP.

IV. SUMMARY

The 3B20D/DMERT system has taken a significant departure from earlier switching processors in many areas, but perhaps none is so visible as the craft interface. The use of flexible video displays makes it possible to adapt the 3B20D to diverse applications quickly and economically. Also, the introduction of a reliable, secure, high-capacity data link for remote maintenance makes the 3B20D/DMERT system well suited for unattended operation, with resultant cost savings.